This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

Figure 6A, Figure 6B, Figure 6C, and Figure 6D are illustrations of graphical user interfaces for configuring interconnections between programmable system blocks, according to an embodiment of the present invention.

Please replace the paragraph beginning at page 10, line 5 with the following new paragraph:

Referring now to Figure 1B, a user module placement work-space includes a resource graphic window 360 illustrating the placement of user modules 304 with respect to the available resources (e.g., available programmable system blocks 410 of a microcontroller) in a hardware layout graphical display. Throughout this application the term resource image may denote the blocks 410 upon which user modules 304 are placed in window 360. As the resource images may represent programmable system blocks in one embodiment, the resource images may be referred to as programmable system blocks for convenience. It will be understood that the resource images may represent other resources however, as the present invention is not limited to implementing the user modules 304 in programmable system blocks. Figure 1B shows a number of digital programmable system blocks 410a along the top row (e.g., the blocks labeled DBA00, DBA01, etc.), as well as four columns of analog programmable system blocks 41 0b (e.g., the blocks labeled ACA00, ACA01, etc.). The present invention is well suited to using any number of analog and digital programmable system blocks 410. Furthermore, the blocks in graphic window 360 are not limited to representing programmable system blocks.

Please replace the paragraph beginning at page 10, line 22 with the following new paragraph:

A single user module 304 may map to one or more programmable system blocks 410. Color coding (not shown) may be used to relate the user modules 304 of selected modules window 306 with their schematic placement in resource graphic window 360. The analog 410b and digital 410a

programmable system blocks may be more generally defined as two different classes to which a user module 304 maps. The present invention is well-suited to having many different classes.

Please replace the paragraph beginning at page 11, line 4 with the following new paragraph:

Referring now to Figure 1C, a pin-out configuration work-space is shown. The pin-out configuration work-space allows the user to connect programmable system blocks 410 to input/output (I/O) pins, as well as configure the I/O pins' drive characteristics. In one embodiment, a pin configuration window 380 may be used to configure pins. Pin configuration window 380 has a port column 381, a select column 382, and a drive column 383. In another embodiment, a user may set pin configurations by clicking on the GUI of the chip 610. The operation of these features will be discussed more fully herein.

Please replace the paragraph beginning at page 12, line 17 with the following new paragraph:

Referring now to step 230 of Figure 2, in response to a request from a user for a potential (e.g., valid) position for a selected user module 304, a position is computed. The computer automatically determines the possible placements based on the available resources and the number of programmable system blocks 410 and the types of programmable system blocks 410 that are required for the unplaced user module 304. Because the user does not need to determine the potential placements, designing the circuit is faster and less error prone than conventional methods which do not provide such guidance.

Please replace the paragraph beginning at page 13, line 2 with the following new paragraph:

User modules 304 may require multiple programmable system blocks 410 to be implemented. In some cases, user modules 304 may require special ports or hardware which may limit which programmable system blocks 410 can be used for their implementation. The process

of mapping a user module 304 to programmable system blocks 410, such that the user module 304 is realized within the microcontroller, may be referred to as "user module placement." An embodiment automatically determines the possible placements of a user module 304 based on an Extensible Markup Language (XML) user module description and the hardware description of the underlying chip. However, the present invention is not limited to using XML descriptions. The potential placement positions may be automatically inferred based on the XML input data. Therefore, the placement process of embodiments of the present invention is data driven.

Please replace the paragraph beginning at page 13, line 15 with the following new paragraph:

In step 240, one or more programmable system blocks 410 are highlighted to indicate a possible position for the user module 304 based on, for example, XML input data. The placement is shown in a graphical hardware layout diagram 360 by highlighting the programmable system blocks 410 involved. For example, referring to Figure 1B, the ADCINC12_1 user module 304 has been selected for placement in the window 360. This user module 304 requires two digital blocks 410a and one analog block 410b. The digital programmable system blocks 410a labeled DBA00 and DBA01 are highlighted to indicate a possible position for the ADCINC12_1 user module 304. Referring now to Figure 3A, the analog programmable system block 410b labeled ASB20 is highlighted to indicate that it is a possible position for the analog portion of the ADCINC12_1 user module 304. Embodiments may use color coding to associate the highlighting color with a unique color assigned to that user module 304.

Please replace the paragraph beginning at page 14, line 4 with the following new paragraph:

User module placement is described in co-pending US patent application serial number 09/989,762, filed concurrently herewith, entitled "A SYSTEM AND METHOD FOR PERFORMING NEXT PLACEMENTS AND PRUNING OF DISALLOWED PLACEMENTS

FOR PROGRAMMING AN INTEGRATED CIRCUIT," by Ogami et al., attorney docket number CYPR-CD01175M and assigned to the assignee of the present invention and incorporated herein by reference.

Please replace the paragraph beginning at page 14, line 11 with the following new paragraph:

Referring now to Figures 3A-3C and to step 250 of Figure 2, after placing a user module 304, a user may desire to move it to another programmable system block 410 (or blocks). In step 250, a new possible position for a user module 304 is computed, in response to a user request for a new position for the user module 304. The user may select a next position button 371 to cause this to occur. Figures 3A-3C illustrate three possible positions for the analog portion of the ADCINC12_1 user module 304. The user may then click on a place module button 372 to place the module 304. Placements that are incompatible with the user module requirements (e.g., characteristics) are automatically pruned out by the software and therefore are not displayed as valid placements. In one embodiment, all positions are shown to the user, sequentially, each time the next placement icon 371 is selected. However, if a potential placement involves a programmable system block 410 that has already been used (e.g., by another placed user module 304), then in these cases the place user module icon 372 is grayed out indicating that this placement is only valid if the resources were vacant. This allows the user to see all possible placements.

Please replace the paragraph beginning at page 15, line 24 with the following new paragraph:

User module next placement is described in co-pending US patent application serial number 09/989,781, filed concurrently herewith, entitled "SYSTEM AND METHOD FOR DECOUPLING AND ITERATING RESOURCES ASSOCIATED WITH A MODULE," by Ogami et al., attorney docket number CYPR-CD01180M and assigned to the assignee of the present invention and incorporated herein by reference.

Please replace the paragraph beginning at page 16, line 6 with the following new paragraph:

Steps 210 through 250 may be repeated to allow the user to add more user modules 304. Each time a new user module is selected, a system resource window may be updated. Referring again to Figure 1A, for each user module 304 selected, the system updates the data in the Resource Manager window 350 with the number of occupied programmable system blocks 410, along with RAM and ROM usage used by the current set of "selected" user modules 304. The system may also prevent a user from selecting a user module 304 if it requires more resources than are currently available. Tracking the available space and memory of configurations for the design may be performed intermittently during the whole process of configuring the microcontroller. Embodiments provide a live graph tracking the programmable system blocks 410 used by percentage. The RAM and ROM monitors may track the amount of RAM and ROM required to employ each selected user module 304.

Please replace the paragraph beginning at page 16, line 19 with the following new paragraph:

After the user has selected one or more user modules 304, the user may select global parameters and user module parameters such as, for example, the gain of an amplifier, a clock speed, etc. Referring now to Figure 4 and to step 260 of Figure 2, in response to a user clicking on a region on a programmable system block 410 an interface 510 is displayed which allows the setting of user module parameters. For example, the user may place "the cursor" over the lower-left corner of a programmable system block 410 to set input parameters. The system may display a superficial chip or a changed cursor in response to this. The user may then left-click a mouse, for example, to bring up a user module parameter window 510 to configure the user module input parameters. The process may be repeated in the lower-right corner of the programmable system block 410 for output parameters and on the upper-left corner for clock parameters. The present invention is not limited to these steps for bringing up a user module pop-up window 510, however. The system may then

display the selected parameters in a user module parameter window 520. Various pop-up windows may be data driven in that the contents of the pop-up window may depend on, for example, the user module 304 selected. Alternatively, user parameters may be set in the user module parameter window 520.

Please replace the paragraph beginning at page 17, line 12 with the following new paragraph:

When the user module 304 is placed (e.g., instantiated) on a particular programmable system block 410 the register settings and parameter settings may be mapped to a physical register address on the chip. This may also associate interrupt vectors that the user module 304 uses based on the programmable system block 410. Each of the digital blocks 410a maps to one vector and each column of analog blocks 410b maps to one vector. Once the user modules 304 are placed and the parameters are set, all the physical address registers that are associated with that user module 304 are fixed and the register values are determined.

Please replace the paragraph beginning at page 19, line 7 with the following new paragraph:

Pin configuration is described in co-pending U.S. patent application serial number 10/032,986, filed October 29, 2001, entitled "PIN-OUT CONNECTIONS/DRIVE LEVELS DIRECT-SET BY DROP DOWN LIST," by Ogami et al., attorney docket number CYPR-CD01173M and assigned to the assignee of the present invention and incorporated herein by reference.

Please replace the paragraph beginning at page 19, line 13 with the following new paragraph:

Referring now to Figure 6A-6D and to step 280 of Figure 2, embodiments provide many different windows to assist the user in setting various parameters to 1 specify interconnectivity of programmable system blocks 410. Referring to Figure 6A, the user may cause window 605 to

appear to configure the analog output buffer. Referring to Figure 6B, the user may cause a clock window 606 to appear by clicking on a clock MUX 616 to configure which clock will be the input to a column of analog programmable system blocks 410b. Referring to Figure 6C a port selection window 607 is shown. The port selection window 607 may be made to appear by clicking on or near the pin input MUX 608. The user may then select the input port. Referring now to Figure 6D, the user may click on or near the analog clocking MUX 614 to cause a window 613 to appear to select which digital programmable system block 410a should be selected by the clock MUX (616 of Figure 6B).

Please replace the paragraph beginning at page 20, line 16 with the following new paragraph:

Embodiments automatically generate source code files for realizing the user modules within the programmable system blocks 410 in the chip. Throughout this application the term source code may be defined as the code that is used to program registers on the chip to implement the selected user modules 304 as they have been placed by the user and to configure the programmable system blocks to operate with the user selected parameters, interconnections, and pin-outs. Thus, automatically generated source code programs may realize the user modules 304 within the programmable system blocks 410.

Please replace the paragraph beginning at page 21, line 1 with the following new paragraph:

The automatically generated files may be programmed into flash memory of a target device (e.g., a microcontroller). The flash memory may then be used to program registers on the target device in order to implement a particular user module 304 in hardware (e.g., actual programmable system blocks 410). Therefore, the source code may be generated based on the selection, placement, and configuration of the user modules 304. For example, the automatic code generation processes

take into account the parameterization of the user modules 304 and the placement of the user modules 304, which the user may perform via a GUI of embodiments of the present invention.

Please replace the paragraph beginning at page 21, line 23 with the following new paragraph:

Automatic generation of source code files is described in co-pending US patent application serial number 09/998,848, filed November 15, 2001, entitled "DESIGN SYSTEM PROVIDING AUTOMATIC SOURCE CODE GENERATION FOR PERSONALIZATION AND PARAMETERIZATION OF USER MODULES," by Ogami, attorney docket number CYPR-CD01177M and assigned to the assignee of the present invention and incorporated herein by reference.

Please replace the paragraph beginning at page 22, line 7 with the following new paragraph:

After the microcontroller has been configured to implement the selected user modules 304, the user may wish to program desired functionality into the microcontroller. For example, a user may wish the user modules 304, now implemented in programmable system blocks in hardware, to function in a certain fashion when the microcontroller is being used. Embodiments automatically generate APIs, which can be used to perform common functions that are required to interact with the user module (e.g., how to start the timer, how to stop the timer, how to talk to the timer, etc.) from an application program level. For example, a user may insert an API into a software program which he writes. Thus, one type of API may be described as a function call; however, APIs are not limited to function calls.

Please replace the paragraph beginning at page 23, line 6 with the following new paragraph:

Automatic generation of API files may take into consideration the configuration the user made. For example, the values to which certain registers are set may depend on which block 410

the user module 304 is placed in. Figure 9A and Figure 9B illustrate an exemplary API file that is automatically generated for a user module ADCINC12_1. This file contains the actual code for the API. Figure 10 and Figure 11 illustrate two more exemplary API files that may be generated for a user module named ADCINC12_1. Each instantiation of each user module 304 may have such files automatically generated for it. The file in Figure 10 may be used to allow programs written in the C programming language to access the user module APIs. The file of Figure 11 contains equates that are used by the APIs. The APIs simplify the designer's task by declaring values for various registers based on the programmable system blocks 410 the user module 304 occupies.

Please replace the paragraph beginning at page 24, line 4 with the following new paragraph:

Automatic generation of APIs is described in co-pending US patent application serial number 09/994,599, filed concurrently herewith, entitled "AUTOMATIC API GENERATION TO FUNCTIONAL PROGRAMMABLE SYSTEM BLOCKS," by Ogami et al., attorney docket number CYPR-CD01198M and assigned to the assignee of the present invention and incorporated herein by reference.

Please replace the paragraph beginning at page 27, line 9 with the following new paragraph:

Automatic generation of datasheets is described in co-pending US patent application serial number 09/994,600, filed concurrently herewith, entitled "SYSTEM AND METHOD FOR DYNAMICALLY GENERATING A CONFIGURATION DATASHEET," by Ogami et al., attorney docket number CYPR-CD01174M and assigned to the assignee of the present invention and incorporated herein by reference.

AMENDMENTS WITH CHANGES SHOWN: